Word2Vec-ACV：OOV 语境含义的词向量生成模型 *

王永贵，郑　泽†，李　玥

(辽宁工程技术大学 软件学院, 辽宁 葫芦岛 125105)

摘　要：针对 Word2Vec 模型生成的词向量缺乏语境的多义性以及无法创建集外词（OOV）词向量的问题，引入相似信息与 Word2Vec 模型相结合，提出 Word2Vec-ACV 模型。该模型首先基于连续词袋（CBOW）和 Hierarchical Softmax 的 Word2Vec 模型训练出词向量矩阵即权重矩阵；然后将共现矩阵进行归一化处理得到平均上下文词向量，再将词向量组成平均上下文词向量矩阵；最后将平均上下文词向量矩阵与权重矩阵相乘得到词向量矩阵。为了能同时解决集外词及多义性问题，将平均上下文词向量分为全局平均上下文词向量(Global ACV)和局部平均上下文词向量(Local ACV)两种，并对两者取权值组成新的平均上下文词向量矩阵。将 Word2Vec-ACV 模型和 Word2Vec 模型分别进行类比任务实验和命名实体识别任务实验，实验结果表明，Word2Vec-ACV 模型同时解决了语境多义性以及创建集外词词向量的问题，降低了时间消耗，提升了词向量表达的准确性和对海量词汇的处理能力。

关键词：Word2Vec 模型；词向量；共现矩阵；平均上下文词向量

中图分类号：TP391.1　　doi: 10.3969/j.issn.1001-3695.2017.12.0800

Word2Vec-ACV: word vector generation model of OOV context meaning

Wang Yonggui, Zheng Ze†, Li Yue

(*College of Software Liaoning Technical University*, *Huludao Liaoning* 125105, *China*)

Abstract: The Word2Vec model is a neural network model (NNLM) that converts words in text into a word vector. It is widely used in natural language processing tasks such as emotional analysis, question answering robot and so on. Word vectors generated for the Word2Vec model lacked the ambiguity of context and the inability to create OOV word vectors. Based on the similarity information of document context and Word2Vec model, this paper proposed a word vector generation model that conforms to the meaning of OOV context. It is called the Word2Vec-ACV model. The model was similar to the process of the word vector generated by the Word2Vec model, but it was different. First of all, Word2Vec model of the continuous word bag (CBOW) and the Hierarchical Softmax trained the word vector matrix, namely the weight matrix. Secondly, the co-occurrence matrix was normalized to get the average context word vector. Then, the word vector consisted of an average context word vector matrix. Finally, the vector matrix of the average context word vector matrix and the weight matrix were multiplied to get the word vector matrix. In order to simultaneously solved the ambiguity problem of out of vocabulary words and out of vocabulary words to create. In this paper, the average context word vectors were divided into two kinds: the global average context word vector (global ACV) and the local average context word vector (local ACV) . In addition, the two taken the weight value to form a new average context word vector matrix. The Word2Vec model can effectively express the word in vector form. Experiments on analogical tasks and named entity recognition (NER) tasks respectively, the results show that the Word2Vec-ACV model is superior to the Word2Vec model in the accurate expression of the word vector. It is a word vector representation method to create a contextual context for OOV words.

Key words: Word2Vec model; Word Vector; The co-occurrence matrix; ACV

在自然语言处理领域中，表征学习（representation learning）是指从单个或一组符号中学习其赋予的含义或其指代的事物。表征学习的主要内容是词汇学习(vocabulary learning)，如何将词汇中隐藏的信息以词向量的形式表达出来已经成为学术界和工业界普遍研究的热点。伴随对深度学习[1]的深入研究，在监督学习任务中，将神经网络语言模型（NNLM）训练出来的词向量作为文本特征，与简单且标准化的词袋模型（BOW）[2, 3]映射生成的词向量作为文本特征相比，有显著地提高。

收稿日期：2017-12-09；**修回日期：**2018-01-19　**基金项目：**国家自然科学基金青年基金资助项目（61404069）

作者简介：王永贵（1967-），男，内蒙古赤峰人，教授，硕士，主要研究方向为大数据、数据挖掘；郑泽（1991-），男（通信作者），硕士，主要研究方向为大数据、数据挖掘（zhengze@aliyun.com）；李玥（1993-），女，硕士，主要研究方向为智能信息处理.

词袋模型将文本看成由词组成的集合，忽略词序、语法和句法等信息，集合中各词都是相互独立，不受其他词出现地影响。词袋模型将文本映射成与训练集相同维度的向量，各向量的分量值分别表示该分量所对应的词在文本中出现的次数。在传统分类器上词袋模型有很好的分类效果，但随着新词的增加，向量的维数也会随之增加，这样会导致维数灾难现象地产生。

在 Bengio 等人[4]提出三层神经网络语言模型的基础上 Mikolov 等人[5,6]于 2013 年首次提出 Word2vec 模型，该模型仅考虑“局部上下文”来学习有意义的词向量，得益于浅层的神经网络结构，使得其可以从大型的语料库中有效地训练出词向量。然而 Word2vec 模型只能从给定的语料库中训练出词向量。假如在任务中，遇到一个在训练过程中没有出现过的新单词，就必须重新使用 Word2vec 模型为这个新词单独创建词向量，这就导致大量重复的时间消耗在模型的训练上。此外，单一的词向量并不能最优地表示一个具有多重含义的词。例如，“包袱”既是指用布包起来的包儿，也是比喻某种负担，只有考虑到这个词的局部语境，才能确定恰当的含义。

1　相关工作

过去，NNLM 以多种方式解决了多义性的问题。由于 Word2Vec 模型对文本中单词顺序不敏感，Wang 等人[7]提出了基于 Word2Vec 改进的 Wang2Vec 模型。该模型由结构化的 Skip-gram 模型和连续窗口方法组成，将语序纳入到 Word2vec 中，对语法效果有明显的提高。Reisinger 等人[8]提出 Statistical Multi-Prototype Vector-Space Models of Word Meanig 模型，通过聚合单词出现的上下文信息来编码单词的多重含义。Trask 等人[9]提出 Sense2Vec 模型，是对 Word2Vec 模型的改进，在语料训练的过程中加入词性的标注，生成新表示形式。例如，一个词同时拥有名词和动词两种词性。但是以上模型都不是针对同时解决 OOV 和多义性问题而设计的，并且有些模型需要更多的参数和训练过程中增加额外步骤使得模型变得复杂。针对以上模型不能同时解决 OOV 和多义性问题，本文提出 Word2Vec-ACV 模型。

Word2vec-ACV 首先把整个语料库放入到 Word2Vec 模型中训练出权重矩阵 W，然后将共现矩阵 Co 进行归一化处理得到平均上下文词向量矩阵 S，最后将平均上下文词向量矩阵 S 与权重矩阵 W 相乘得到词的向量表示。应用 Text8 语料库数据集训练生成的词向量分别对 Question Word 类比任务[5,6]和命名实体识别（NER）[10]任务进行实验，实验结果表明，Word2vec-ACV 模型生成词向量的准确性优于 Word2Vec 模型，能有效地创建 OOV 词向量以及赋予词向量多重含义。

2　Word2Vec 及 Word2Vec-ACV 的原理介绍

Word2Vec-ACV 模型是在 Word2Vec 模型的基础上针对词之间的相似特性所提出，通过结合词之间的相似性信息提高词嵌入的准确性，是一种用于生成词向量的神经网络语言模型。

2.1　基于 CBOW 和 Hierarchical Softmax 的 Word2vec 模型

Word2Vec 模型是建立在神经网络语言模型（NNLM）基础上，移除前向反馈神经网络中非线性的隐藏层（hidden layer），直接将中间的嵌入层（embedding layer）与输出层（softmax layer）相连。忽略其上下文中的语序信息，将输入层输入的上下文词向量汇总到嵌入层得到一个连续的词向量 $e \in R^N$，然后直接与 hierarchical softmax 相连得到 Word2vec 模型，如图 1 所示。

输入层包含预测目标词的 c 个上下文的词向量，其中 V 表示词向量的长度。嵌入层先将输入的上下文词向量 $v(w_{i+1}), v(w_{i+2}), \cdots, v(w_{i+c}) \in R^V$ 求和取平均值作为输出得到 $e \in R^N$。其中，e 是一个 N 维向量。输出层对应一颗二叉树，用语料中出现过的词当叶子节点，以各词在语料中出现的次数当权重构造出 Huffman 树。在 Huffman 树中，叶子节点共 $V(=|D|)$ 个，分别对应词典 D 中的词 w（图 1 中标为白色的若干节点），非叶子节点 $V-1$ 个（图 1 中标为黑色的若干节点）。其中，在层次 Softmax 模型中每个非叶子节点对应一个辅助向量 $V'_{(n,j)}$。$L(w_2)$ 表示从根节点到达叶子节点 w_2 的路径长度（图 1 中用黑线标注的线段）。$n(w,j)$ 表示从根节点到目标词 w 路径上的第 j 个节点。

叶子节点上的任意词 w，在 Huffman 树中必存在一条从根节点到目标词 w 对应非叶子节点 $n(w,j)$ 的路径。路径上存在 $L(w)-1$ 个分支，将每一个分支看做一次二分类，每一次二分类就产生一个概率，将概率相乘，得到该模型的目标函数 $p(w=w_O)$。先对目标函数取对数得到似然函数 E，再通过随机梯度算法对似然函数进行迭代，得到最优的参数，从而获得最优的权重矩阵 W。

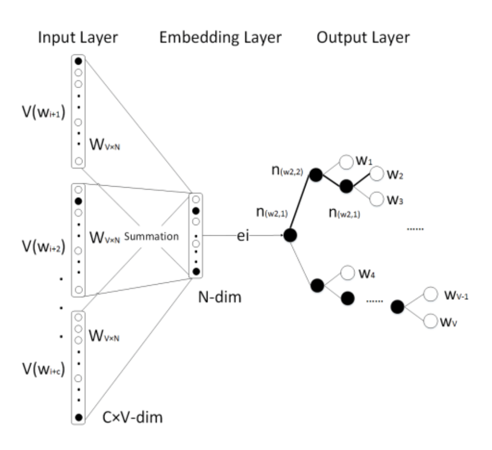


图 1　基于 CBOW 和 Hierarchical Softmax 的 Word2vec 模型

2.2　Word2Vec-ACV 模型

本文有针对性地对一定量的文献进行研究[5~9]，发现根据上下文单词出现的频率，所构成的词向量矩阵 $C \in R^{V \times N}$ 能计算出词典 D 中 V 个词相互之间的相似度，得到由相似度为 0 到 1 之间的一个值所构成的相似矩阵 $Sim \in R^{V \times V}$ 或者词 w 的相似向量 $Sim_w \in R^V$。这些相似性将会保留在词向量中。例如，相似的两个词之间的词向量的余弦值接近 1，或者词向量矩阵 $C \in R^{V \times N}$ 的标量积近似于相似矩阵 $Sim \in R^{V \times V}$。显然，获得词向量矩阵 $C \cdot C^T \approx Sim$ 最优的方式就是计算相似矩阵 $Sim \in R^{V \times V}$ 的奇异值

分解（SVD），并使用对应于 N 最大特征值的特征向量[11, 12]。由于语料库中数以万计的词组成的相似矩阵是非常巨大的，但大多数词都不是同义词，所以相似矩阵是稀疏矩阵便于进行 SVD 计算。

Word2Vec 模型的输入层是将与目标词相连的 K 个上下文词作为输入，经过嵌入层，将中间结果沿着输出层 Huffman 树上的非叶子节点到达目标词所在的叶子节点，此时选择的 K 个词与目标词的相似度较大，如果随机选择 K 个词，则与目标词之间的相似度接近于 0。因此，一个词的词向量 $c_w \in R^N$ 与所有词构成的词向量矩阵 $C \in R^{V\times N}$ 相乘得到一个向量 $\hat{v}_w \in R^V$ 的相似度接近目标词 v_w。基于 CBOW 和 Hierarchical Softmax 的 Word2vec 模型在训练词向量的过程中能很好地解释这一相似性过程。在训练过程中，每个出现的词 w_i 是以一个二进制向量 $v_{w_i} \in R^V$ 的形式通过输入到模型中与权重矩阵 $W \in R^{V\times N}$ 权重矩阵相乘得到 e_i，然后经过 Hierarchical Softmax 层抵达叶子节点为 w_i 的目标节点，说明 e_i 包含有词 w_i 与其他词的相似信息。可以很好地解释一个词的词向量 $c_w \in R^N$ 与所有词构成的词向量矩阵 $C \in R^{V\times N}$ 相乘得到一个向量 $\hat{v}_w \in R^V$ 的相似度接近目标词 v_w。

将相似性信息保留在词向量中，会在一定程度上提升词的嵌入效果，然而 Word2Vec 模型并没有考虑将相似信息纳入到该模型中。本文提出的 Word2Vec-ACV 模型是将相似性信息纳入 Word2Vec 模型所创建。Word2Vec-ACV 模型如图 2 所示。

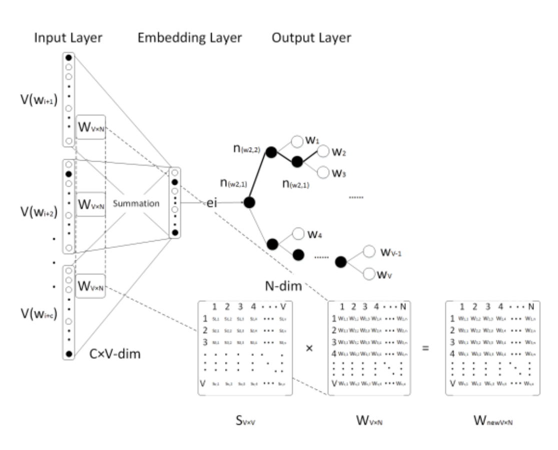


图 2　ACV-Word2vec 模型

Word2Vec-ACV 模型与 Word2Vec 模型的词向量生成过程类似，但细节不同。Word2Vec-ACV 首先将语料通过 Word2Vec 训练出词向量的权重矩阵 $W \in R^{V\times N}$，再将由共现矩阵 $Co \in R^{V\times V}$ 进行归一化处理得到的平均上下文词向量 $\bar{v}_{w_i} \in R^V$ 组成得到平均上下文词向量矩阵 $S \in R^{V\times V}$，最后将权重矩阵 $W \in R^{V\times N}$ 与经过归一化处理后得到的平均上下文词向量矩阵 $S \in R^{V\times V}$ 相乘得到新的权重矩阵 $W_{new} \in R^{V\times N}$。其中，平均上下文词向量矩阵 $S \in R^{V\times V}$ 构造过程如下：首先是由第 i 次出现的词 w 与词 w 在同一个窗口出现的词所构成的二进制向量 $v_{w_i} \in R^V$，其中上下文词出现的位置为 1，其余的位置为 0。对每个词每次的出现结果累加得到共现矩阵 $Co \in R^{V\times V}$。再对共现矩阵 $Co \in R^{V\times V}$ 每一行除以词 w_i 在文本中出现的次数 M 得到平均上下文词向量矩阵 $S \in R^{V\times V}$，其中每一行代表词 w_i 的平均上下文词向量 $\bar{v}_{w_i} \in R^V$。

为了能够快速、高效地为集外词创建词向量以及使创建出来的词向量符合上下文语境，本文将平均上下文词向量分为全局平均上下文词向量（global ACV）和局部平均上下文词向量（local ACV）组成两种。其中 global ACV 是对整个语料库中的词 w_i 的词向量根据词 w_i 出现的次数 $M_{w_{global}}$ 求平均值，即由语料生成的共现矩阵 Co_{global} 中的每一行除以该词在文本中出现的次数 $M_{w_{global}}$ 获得，记为 $V_{w_{global}}$。

$$V_{w_{global}} = \frac{1}{M_{w_{global}}}\sum_{i=1}^{M_{w_{global}}} v_{w_i} \tag{1}$$

同理，local ACV 是对集外词所在的文档中词 w_i 的词向量根据词 w_i 出现的次数 $M_{w_{local}}$ 求平均值，记为 $V_{w_{local}}$。

$$V_{w_{local}} = \frac{1}{M_{w_{local}}}\sum_{i=1}^{M_{w_{local}}} v_{w_i} \tag{2}$$

对 $V_{w_{global}}$ 与 $V_{w_{local}}$ 中相同词 w_i 对应的词向量按一定的比例 a 求和得到加权平均上下文词向量矩阵 S，如式(3)所示。

$$S = \left(a \cdot V_{w_{global}} + (1-a)\cdot V_{w_{local}}\right)^T \tag{3}$$

将式(3)中得到的加权平均上下文词向量矩阵 S 与权重矩阵 W 相乘最终得到词向量矩阵 W_{new}。

$$W_{new} = S \times W \tag{4}$$

其中：$a \in [0,1]$，a 的设置用于调节词 w 受上下文影响的程度，这有助于解决语境多义性问题[13]。

虽然该模型与 Word2Vec 模型训练词向量的过程相同，但训练完成后，词向量的计算方式存在差异。在 Word2Vec 的情况下，词向量是调整后权重矩阵 W 的行向量。Word2Vec-ACV 模型是将权重矩阵 W 与经过归一化处理后得到的平均上下文词向量矩阵 $S \in R^{V\times V}$ 相乘得到的新权重矩阵 $W_{new} \in R^{V\times V}$ 的行向量表示词的向量。

3　Word2Vec-ACV 模型推导及其实现

本文对 Word2Vec-ACV 模型的参数推导时将采用随机梯度法（stochastic gradient method），简称 SG 法。因为 SG 法对参数的推导过程实现简单且高效，在 Word2Vec-ACV 模型中输入的词向量 v_w 是已知的，而权重 W 和嵌入层的输出向量 ei 以及 Hierarchical Softmax 模型中的辅助向量 $V_n^{'}(w,j)$ 是未知的，这就需要采用 SG 法对参数进行优化。

3.1　输入层到嵌入层的推导过程

输入层输入的是上下文词向量的平均值，其中每个词用独热编码向量（one-hot encoded vector）[14]表示，即将给定的上下文词表示成 $v_w\{x_1, x_2, \cdots, x_v\}$ 的向量形式，其中向量的分量 x_i 中只有一个为 1，其他分量全为 0。权重矩阵 W 的每一行可理解为输入词 w 的 N 维词向量 $v_w \in R^N$。计算嵌入层输出时，将输入上下文词向量的平均值作为输出，即为 $e \in R^N$。则有

$$e=\frac{1}{C}\left(v_{w_1}+v_{w_2}+\cdots+v_{w_c}\right)^T \tag{5}$$

其中：C 是上下文词向量的个数；$w_1,w_2,\cdots,w_c$ 是上下文中的词；v_w 是词 w 的输入向量。

将输入词向量更新公式定义为

$$v_{w_{I,c}}^{(new)}=v_{w_{I,c}}^{(old)}-\frac{1}{C}\cdot\eta\cdot EH^T \tag{6}$$

其中：$c=1,2,\cdots,C$；$v_{w_{I,c}}$ 是上下文中输入的第 c 个词的向量；η 是学习率；$EH=\frac{\partial E}{\partial e}$ 是对数似然函数对输出向量 e 的导数。

3.2　嵌入层到输出层的推导过程

神经网络语言模型的目标函数通常是取条件概率函数的对数似然函数，最关键的是对条件概率函数 $p(w=w_o)$ 的构造。在 Hierarchical Softmax 模型中，将目标词 w 的概率输出作为条件概率函数。定义如下：

$$p(w=w_O)=\prod_{j=1}^{L(w)-1}\sigma\left(\left\|n(w,j+1)=ch(n(w,j))\right\|\cdot {v'_{n(w,j)}}^T\cdot e\right) \tag{7}$$

其中：$ch(n)$ 是节点 $n(w,j)$ 的左孩子节点；$V'_{n(w,j)}$ 是非叶子节点 $n(w,j)$ 的辅助向量；e 是嵌入层的输出值；$\|x\|$ 是一个特殊函数，在 Huffman 树中用于给非叶子节点的左右孩子节点指定一个类别，即哪个是正类（标签为 1），哪个是负类（标签为-1）。定义为

$$\|x\|=\begin{cases}1 & if\ \ x\ is\ true;\\ -1 & otherwise.\end{cases} \tag{8}$$

通过预测目标词 w2 说明推导过程。在图 2 中，从根节点出发到达叶子节点 w2，中间共经历了 4 次分支，每次分支都可视为进行了一次二分类。使用 $\|x\|$ 函数对节点进行标注，其中，左孩子节点的标签为 1，右孩子节点标签-1。根据逻辑回归知识，利用 Sigmoid 函数，可计算出分到左孩子节点的概率 $p(n,left)$。

$$p(n,left)=\sigma\left({v'_n}^T\cdot e\right) \tag{9}$$

则分到右孩子节点的概率 $p(n,right)$为

$$p(n,right)=1-\sigma\left({v'_n}^T\cdot e\right)=\sigma\left(-{v'_n}^T\cdot e\right) \tag{10}$$

从根节点到叶子节点 w_2 的路径可得到 w_2 作为词输出的条件概率 $p(w_2,w_o)$，将式(9)(10)代入到 $p(w_2,w_o)$ 中，计算公式如式(11)和(12)所示。

$$p(w_2=w_O)=p(n(w_2,1),left)\cdot p(n(w_2,2),left)\cdot p(n(w_2,3),right) \tag{11}$$

$$p(w_2=w_O)=\sigma\left({v'_{n(w_2,1)}}^T\cdot e\right)\cdot\sigma\left({v'_{n(w_2,2)}}^T\cdot e\right)\cdot\sigma\left(-{v'_{n(w_2,3)}}^T\cdot e\right) \tag{12}$$

经归一处理得到

$$\sum_{i=1}^{V}p(w_i=w_O)=1 \tag{13}$$

以下开始推导非叶子节点对应的辅助向量 $V'_{n(w,j)}$ 的参数更新方程，为下文梯度函数的推导方便，将式(7)中的符号‖‖里的内容简记为 $\|\bullet\|$，将 $V'_{n(w,j)}$ 简记为 v'_j，如式(14)(15)所示。

$$\|\bullet\|:=\|n(w,j+1)=ch(n(w,j))\| \tag{14}$$

$$v'_j:=v'_{n_{w,j}} \tag{15}$$

一般基于神经网络语言模型的目标函数通常取对数似然函数，即如式(16)所示。

$$E=\sum_{w\in c}\log p(w=w_O\mid w_I) \tag{16}$$

将式(7)代入对数似然函数式(16)中得到目标函数 E，即如式(17)所示。

$$E=-\log p(w=w_O\mid w_I)=-\sum_{j=1}^{L(w)-1}\log\sigma\left(\|\bullet\|{v'_j}^T e\right) \tag{17}$$

式(17)推导出的对数似然函数 E 就是 CBOW 模型的目标函数。为了使目标函数 E 的值最大，式（18）采用随机梯度上升算法，梯度类算法的关键是给出梯度计算公式，下文为梯度计算的推导过程。

随机梯度上升算法的过程是每对样本进行一次采样，就会对目标函数中的相关参数进行一次更新。由目标函数 E 式(17)可知，该函数参数包括向量 e，v'_j，$w\in c$，$j=1,\cdots,L(w)-1$。首先考虑 E 关于 $v'_j e$ 的梯度计算，即

$$\frac{\partial E}{\partial v'_j e}=\left(\sigma\left(\|\bullet\|v'_j e\right)-1\right)\|\bullet\|=\begin{cases}\sigma\left({v'_j}^T e\right)-1 & (\|\bullet\|=1)\\ \sigma\left({v'_j}^T e\right) & (\|\bullet\|=-1)\end{cases}=\sigma\left({v'_j}^T e\right)-t_j \tag{18}$$

其中：如果 $\|\bullet\|=1$ 则 $t_j=1$，否则 $t_j=-1$。

接下来考虑 E 关于非叶子节点 $n(w,j)$ 的辅助向量 v'_j 的梯度计算，即

$$\frac{\partial E}{\partial v'_j}=\frac{\partial E}{\partial v'_j e}\cdot\frac{\partial v'_j e}{\partial v'_j}=\left(\sigma\left(v'_j e\right)-t_j\right)\cdot e \tag{19}$$

于是，v'_j 的更新公式可写为

$${v'_j}^{(new)}={v'_j}^{(old)}-\eta\left(\sigma\left({v'_j}^T e\right)-t_j\right)\cdot e \tag{20}$$

其中：η 表示学习率；$\sigma\left({v'_j}^T e\right)-t_j$ 表示非叶子节点 $n(w,j)$ 的预测误差；$t_j=1$ 表示接下来到左孩子节点；$t_j=0$ 表示接下来到右孩子节点；$\sigma\left({v'_j}^T e\right)$ 是预测值，是预测非叶子节点接下来是左孩子节点还是右孩子节点的概率值。

3.3　嵌入层到输入层的推导过程

通过对数似然函数反向推导出权重矩阵 W 的更新公式，考虑 E 关于嵌入层的输出向量 e 的梯度计算式(21)，即

$$\frac{\partial E}{\partial e}=\sum_{J=1}^{L(w)-1}\frac{\partial E}{\partial v'_j e}\cdot\frac{\partial v'_j e}{\partial e}=\sum_{j=1}^{L(w)-1}\left(\sigma\left({v'_j}^T e\right)-t_j\right)\cdot v'_j:=EH \tag{21}$$

将式(21)直接代入式(6)可得到权重矩阵 W 的更新公式。

3.4　权重矩阵 Wnew 的推导过程

将更新后的权重矩阵 W 与平均上下文词向量矩阵 $S\in R^{V\times V}$

相乘得到新的权重矩阵 $W_{new} \in R^{V\times N}$ 作为词的向量表示。

$$W_{new} = S \times W \tag{22}$$

4 实验

4.1 实验环境

实验环境为 Pentium Dual-Core CPU E5300{@}2.60 GHz，2 GB 内存、500 GB 硬盘的台式机。操作系统为 Windows 7，实验工具为 Anaconda2(64-bit)以及 JetBrains PyCharm Community Edition 2017.1.2 x64

4.2 数据集的获取

类比任务中使用到的 text8 和 questions-words 数据集均是从 Google 官网下载得到，语料库 text8 是由 10{8}个单词组成的一行句子，其中包含 27 种字符，即小写的从 a 到 z 的字母及空格符。数据集 questions-words 中包含 family 等 14 个类别，每个类别中的数据是 4 列。

命名实体识别任务中使用的数据是 CoNLL-2003 命名实体[10]任务中的数据。其中一份用于训练 Word2Vec 词向量的 Traing 数据，以及两份用于测试的 Development 数据和 Testa、Testb 数据。数据中每行包含 4 个字段：单词、POS 标签[15]、块标签、命名实体标签。用 O 标记的是外来的命名实体，I-XXX 标签用于 XXX 类型的命名实体中的单词。数据包含 4 种类型的实体：PER、ORG、LOC、MISC。

4.3 两组实验任务

为了有效地说明 Word2Vec-ACV 模型的优点，本文分别通过类比任务（analogy task）和命名实体识任务（NER task）对 Word2Vec-ACV 模型进行评估实验。

4.3.1 类比任务

为了表明 Word2Vec-ACV 模型创建的词向量能有效地反映出词之间的语义的关系，本文借用 Mikolov 等人在 Word2Vec 论文中提到的类比任务[6, 7]进行评估实验。本实验首先运用基于 CBOW 和 Hierarchical Softmax 的 Word2vec 模型对 text8 语料库进行训练，其中式(3)中的 a 赋值为 1，$hs=1$，嵌入维度 200 维，随机种子为 3，且上下文窗口取值为 5。然后将从含有 17 005 207 个单词（删除掉那些计数<5 的单词）的语料库中训练出来的 253 854 个 Word2Vec 词向量和再将 Word2Vec 词向量与这 253 854 个词相应的平均上下文词向量矩阵相乘得到的 Word2Vec-ACV 词向量运用到类比任务中。其中，类比任务是一系列 $A-B=C-D$ 类问题。即通过类比词向量 A 减去词向量 B 的形式预测出词向量 C 对应的词向量 D 然后再与正确答案进行匹配，从而评估出模型性能。例如，给定 A：King、B：Queen、C：Man 预测 D：Woman。

为了评估 Word2Vec 模型和 Word2Vec-ACV 模型在类比任务中的准确性，引入准确性[16]计算式(23)。

$$Accuracy = \frac{correct}{correct + incorrect} \tag{23}$$

其中：$correct$ 表示预测正确的个数；$incorrect$ 表示预测错误的个数。实验结果如表 1 所示。

表 1 Word2Vec 和 Word2Vec-ACV 模型 1 次迭代类比任务准确性/%

类别	Word2Vec	Word2Vec-ACV
capital-common-countries	10.5	36.0
capital-world	4.3	19.9
currency	4.5	11.6
city-in-state	10.9	20.7
family	53.9	61.4
gram1-adjective-to-adverb	6.1	8.6
gram2-opposite	15.4	18.3
gram3-comparative	44.9	44.3
gram4-superlative	22.9	20.4
gram5-present-participle	11.9	17.3
gram6-nationality-adjective	36.0	37.6
gram7-past-tense	17.3	18.5
gram8-plural	27.9	29.5
gram9-plural-verbs	23.5	20.2
total	20.5	25.7

表 1 列出了 Word2Vec 和 Word2Vec-ACV 模型对 14 个类别下进行 1 次迭代各自的准确性。在前 3 个类比任务中，Word2Vec-ACV 模型的准确性都是 Word2vec 模型的 2 倍以上，其中 capital-common-countries 类别任务中的准确性高达 3.43 倍。通过对实验结果分析表明 Word2Vec-ACV 模型在类比任务中的准确性要比 Word2Vec 模型在类比任务中的准确性要高。这是由于在某些类比任务中词只有单一含义，而其他类比任务中词有多重含义。说明词义的多样性对准确性计算的影响较大，也表明在获取词之间的语义关系方面，Word2Vec-ACV 模型要优于 Word2vec 模型。

表 2 Word2Vec 和 Word2Vec-ACV 模型 10 次迭代类比任务准确性/%

类别	Word2Vec	Word2Vec-ACV
capital-common-countries	64.8	79.01
capital-world	33.9	57.9
currency	15.9	19.6
city-in-state	29.3	44.3
family	79.5	76.4
gram1-adjective-to-adverb	11.01	16.7
gram2-opposite	24.4	27.3
gram3-comparative	64.9	64.3
gram4-superlative	41.9	38.4
gram5-present-participle	30.9	31.3
gram6-nationality-adjective	71.6	67.6
gram7-past-tense	30.3	33.3
gram8-plural	48.9	49.5
gram9-plural-verbs	41.5	32.2
total	42.1	47.4

表 2 列出了 Word2Vec 和 Word2Vec-ACV 模型对 14 个类别下进行 10 次迭代各自的准确性。实验结果表明 10 次迭代的类比任务的准确率整体优于在 1 次迭代中的准确率，且在 10 次迭代中的 capital-world 类别任务，Word2Vec-ACV 模型训练出的词向量准确率是 Word2Vec 模型的 1.7 倍。通过对实验结果分析表明 Word2Vec-ACV 模型在类比任务中的准确性要比 Word2Vec 模型在类比任务中的准确性要高，表明 Word2Vec-ACV 模型训练出的词向量要优于 Word2Vec 训练出的词向量，并说明语料迭代的次数会影响到实验的结果。

4.3.2 NER 任务

Word2Vec-ACV 模型的主要优点是能够使用局部平均上下文词向量矩阵来创建 OOV 词向量，并区分词的不同含义。实验采用 CoNLL 2003 NER 任务[10]作为外部评估，与常规的 Word2vec 模型相比来说明 Word2Vec-ACV 模型的这一优点，将 CoNLL 2003 NER 任务中的 Traing 数据用于 Word2Vec 模型训练出词向量 T_{cv}，其中在 Development 和一份包含 Testa，Testb 的测试数据中的 OOV 用零向量表示。并将得到的词量 T_{cv} 分别与 Development 和 Test 数据中的平均上下文词向量矩阵相乘创建 Word2Vec-ACV 词向量 D_{cv}、Ta_{cv} 和 Tb_{cv}。将训练出来的词向量同时与逻辑回归分类模型一起使用。其中，为排除其他因素对结果的影响，本实验只使用词向量作为特征信息，不使用例如 POS 标签等其他信息。为了评价本文提出的 Word2Vec-ACV 模型实验结果的质量，将实验得到的结果引入一个评价标准 $F_1-Measure$ [17]。$F_\beta-Measure$ 是一种计算准确率和召回率加权调和平均的统计量。准确率和召回率是广泛应用于信息检索和统计学分类领域的两个度量值，用来评价结果的质量。准确率是检索出相关文档数与检索出的文档总数的比率，衡量的是检索系统的查准率；召回率是指检索出的相关文档数和文档库中所有的相关文档数的比率，衡量的是检索系统的查全率。计算公式如下：

$$F_\beta-Measure=\frac{(\beta^2+1)*Precision*Recall}{(\beta^2*Precision+Recall)} \tag{24}$$

其中：当 $\beta=1$ 时，即为。$Precision$ 是准确率，表示为命名实体识别任务中识别正确的样本数与总样本的比率；$Recall$ 是召回率，表示为命名实体识别任务中识别正确的样本数与实际任务中的总样本数的比率。根据模型和 CoNLL 2003 NER 任务将准确率和召回率定义为如下公式：

$$Precision=\frac{\text{模型中命名实体识别正 确的样本数}}{\text{模型中命名实体识别的 总样本数}} \tag{25}$$

$$Recall=\frac{\text{模型中命名实体识别正 确的样本数}}{\text{模型中命名实体识别的 总样本数}} \tag{26}$$

依据公式，本实验验证过程分别将 Training 生成 Word2Vec 词向量 T_{cv}，测试数据中的 Testa 和 Testb 分别生成的 Word2Vec-ACV 词向量 Ta_{cv} 和 Tb_{cv}，其中式（3）中的 a 赋值为 0.6。将这三种词向量用于 4 种类型的命名实体任务中，得到的各自的 $F_1-Measure$ 值比较分别如图 3~6 所示。

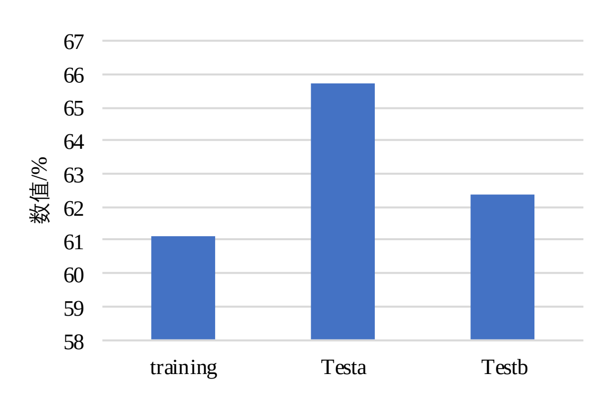


图 3　training、Testa 和 Testb 词向量 LOC 任务中 F_1-Measure 的比较

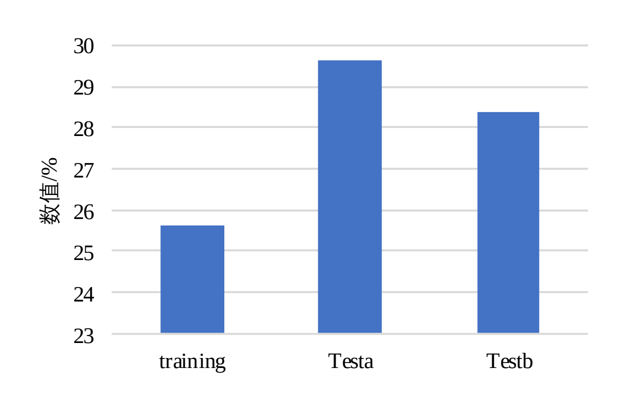


图 4　training、Testa 和 Testb 词向量 MISC 任务中 F_1-Measure 的比较

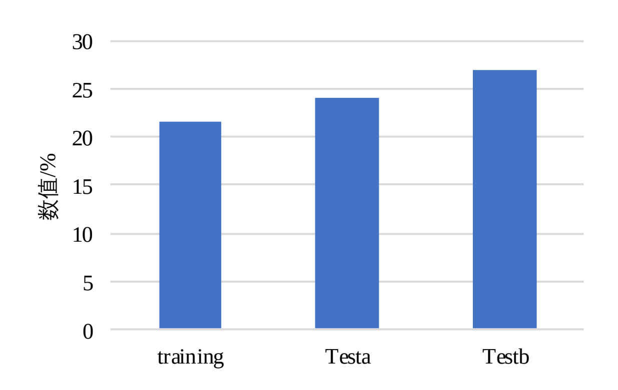


图 5　training、Testa 和 Testb 词向量 ORG 任务中 F_1-Measure 的比较

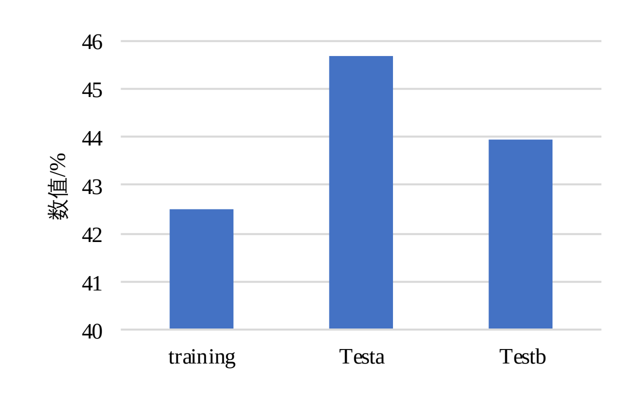


图 6　training、Testa 和 Testb 词向量 FER 任务中 F_1-Measure 的比较

根据图 3~6 的结果显示，用 Training 训练出的 Word2Vec 词向量 T_{cv}、Testa 训练出的 Word2Vec-ACV 词向量 Ta_{cv} 和 Testb

训练出的 Word2Vec-ACV 词向量 Tb_{cv} 与逻辑回归分类模型一起应用于 CoNLL 2003 NER 任务中得到当 a 赋值为 0.6 时，LOC、MISC、ORG 和 FER 这四类命名实体任务任务中的 $F_1-Measure$ 值，在 LOC 命名实体任务中 Training、Testa 和 Testb 的 $F_1-Measure$ 值分别为 61.14、65.75 和 62.4；在 MISC 命名实体任务中 Training、Testa 和 Testb 的 $F_1-Measure$ 值分别为 25.62、29.64 和 28.4；在 ORG 命名实体任务中 Training、Testa 和 Testb 的 $F_1-Measure$ 值分别为 21.55、24.2 和 26.95；在 FER 命名实体任务中 Training、Testa 和 Testb 的 $F_1-Measure$ 值分别为 42.5、45.69 和 43.93。其中 Testa 训练出的 Word2Vec-ACV 词向量 Ta_{cv} 在 LOC、MISC 和 FER 任务的 $F_1-Measure$ 值最高，Training 训练出的 Word2Vec 词向量 T_{cv} 在 LOC、MISC 和 FER 命名实体任务的 $F_1-Measure$ 值最低。其中在 MISC 任务中，利用 Testa 的 Word2Vec-ACV 词向量 Ta_{cv} 得到的 $F_1-Measure$ 值比单纯利用 Training 数据训练出的 Word2Vec 词向量的 $F_1-Measure$ 值高出 4.02 个百分点，利用 Testb 的 Word2Vec-ACV 词向量 Tb_{cv} 得到的 $F_1-Measure$ 值比单纯利用 Training 数据训练出的 Word2Vec 词向量的 $F_1-Measure$ 值高出 2.78 个百分点。

经过对实验结果的分析，采用 Word2Vec-ACV 模型训练出来的词向量 Ta_{cv} 和 Tb_{cv} 在区分词的不同含义方面要优于单纯采用 Word2Vec 模型训练出来的词向量 T_{cv}。

为了验证 Word2Vec-ACV 模型对 OOV 创建词向量有很好的效果，先对 Training 数据使用 Word2Vec 模型进行 100 次迭代训练出词向量，其中 Development 和 Test 数据中的 OOV 词用零向量表示。然后分别对 Training、Development、和 Test 数据训练出来的词向量进行 NER 实验。

实验时，对式(3)中的 a 分别赋值为 0、0.1、0.2、0.3、0.4、0.5、0.6、0.7、0.8、0.9、1。当 $a=1$ 时，分两种情况：第一种是将 Training 数据训练出来的 Word2Vec 词向量 T_{cv} 与 Training 数据的全局平均上下文词向量（global ACV）相乘得到 Global 词向量，如图 7、8 所示的 global；第二种是将 Development 和 Test 数据训练出的 Word2Vec 词向量分别与 Development 和 Test 数据局部平均上下文词向量（local ACV）相乘创建出 OOV 词向量，如图 7、8 所示的 OOV。当 $0\le a<1$ 时，分别将单个 Development、Test 数据的局部平均上下文词向量（local ACV）与 Training 数据的全局平均上下文词向量（global ACV）在 0~1 间取一个权值 a 与词向量 T_{cv} 相乘创建混合型 OOV 词向量。实验采用 $F_1-Measure$ 值进行评估，评估结果如图 7、8 所示。

实验结果显示，使用 Word2Vec-ACV 模型生成的词向量在 NER 任务中要优于 Word2Vec 模型生成的词向量，其中 Training 代表 Word2Vec 模型生成的词向量用于 NER 任务。当 a 赋值为 1 时，Globale 对应的 $F_1-Measure$ 值与 OOV 对应的 $F_1-Measure$ 值相差不大，因为其词向量的计算都是基于它们各自的全局上下文训练出的 Word2Vec 词向量与它们各自的平均上下文词向量相乘得到的；当 $0\le a<1$ 时，混合型 Word2Vec-ACV 生成的词向量对应 $F_1-Measure$ 值随着 a 的取值成在一定的相关性，其中当 a 赋值为 0.6 时，$F_1-Measure$ 值最高，明显高于 a 赋值不为 0.6 时的 $F_1-Measure$ 值，这说明单个文档的 Local ACV 和全语料库的 Global ACV 混合起来的 Word2Vec-ACV 模型创建出来的混合型词向量优于单个文本创建出来的词向量。经过对实验结果的分析说明，将单个文档的 Local ACV 和全语料库的 Global ACV 混合起来创建出混合型的 Word2Vec-ACV 词向量相比于 Word2Vec 模型生成的词向量更能有效地为集外词创建词向量并使词向量符合其上下文语境的含义。

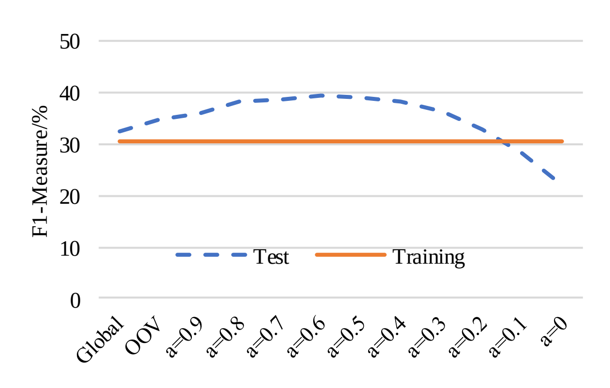


图 7 Test 词向量与 Training 词向量的 F1-Measure 值比较

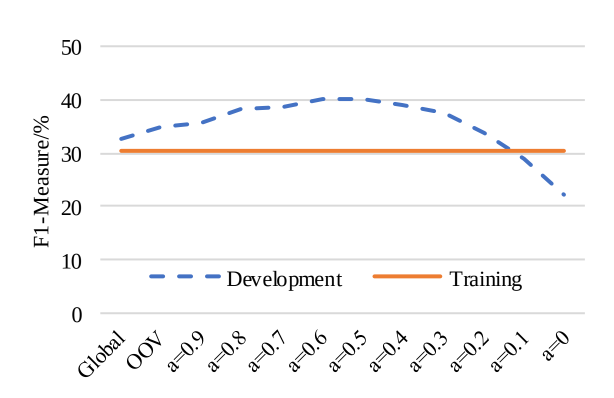


图 8 Development 词向量与 Training 词向量的 F1-Measure 值比较

5 结束语

本文将上下文语境环境的相似性信息纳入到 Word2Vec 模型中，针对传统的自然语言模型不能根据语境环境为集外词创建词向量的问题，本文提出了一种基于平均上下词向量的 Word2Vec-ACV 模型。

该模型基于 CBOW 和 Hierarchical Softmax 框架的 Word2vec 模型训练出权重矩阵 W，再分别对语料库和集外词所在的文档生成共现矩阵 Co_{global} 和 Co_{local}，再对共现矩阵进行归一化处理分别得到语料库的平均上下文词向量矩阵 $V_{wglobal}$ 和集外词所在的文档的平均上下文词向量矩阵 V_{wlocal}。再对 $V_{wglobal}$ 与 V_{wlocal} 中相同词 w_i 对应的词向量按一定的比例 a 求和得到加权平均上下文词向量矩阵 S，最后将平均上下文词向量矩阵 S 与权重矩阵 W 相乘得到最终的词向量矩阵 W_{new}，该模型会在一定程度上提升词的嵌入效果。

将 Word2Vec-ACV 模型与 Word2Vec 模型应用于类比任务

实验和命名实体识别任务中。在类比任务的实验中，Word2Vec-ACV 模型训练出来的词向量应用在 capital-world 类别任务中的准确率是 Word2Vec-ACV 模型训练出来的词向量的 1.7 倍以上，能更加准确地反映出词在特定语境的意义。在命名实体识别任务的实验中，基于 global ACV 和 local ACV 的 Word2Vec-ACV 模型训练出来的词向量得到的 $F_1-Measure$ 值要高于单独使用 global ACV 的 Word2Vec-ACV 模型训练出来的词向量得到的 $F_1-Measure$ 值。其中，当 a 取值为 0.6 时指标 $F_1-Measure$ 的值最高，表明结合 local ACV 创建的 Word2Vec-ACV 词向量在不同语境下能有效地区分词的不同含义。在下一步研究中，将考虑上下文向量矩阵中是否能融入更多元信息，如将语序信息加入到模型中，以期望能更加完善模型生成词向量的准确性。

参考文献：

[1] 张军阳，王慧丽，郭阳，等. 深度学习相关研究综述 [J/OL]. 计算机应用研究, 2018, 35 (7) . [2017-08-17]. http://www. arocmag. com/article/02-2018-07-067. html.

[2] Baeza-Yates R, Ribeiro-Neto B. Modern information retrieval [M]. New York: ACM Press, 1999: 1-5.

[3] Manning C D, Schütze H. Foundations of statistical natural language

[4] processing [M]. Cambridge: MIT Press, 1999: 1-4.

[5] Bengio Y, Ducharme R, Vincent P, *et al.* A neural probabilistic language model [J]. Journal of Machine Learning

[6] Research, 2003 (3): 1137-1155.

[7] Mikolov T, Sutskever I, Chen K, *et al.* Distributed representations of words and phrases and their compositionality [C]// Advances in Neural Information Processing Systems. 2013: 3111-3119.

[8] Mikolov T, Chen K, Corrado G, *et al.* Efficient estimation of word representations in vector space [J]. arXiv preprint, arXiv: 1301. 3781

[9] Wang L, Dye C, Black A W, *et al.* Two//too simple adaptations of Word2Vec for syntax problems [C]// Proc of Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies. 2015.

[10] Reisinger J, Mooney R J. Multi-prototype vector-space models of word meaning [C]// Proc of Conference of the North American Chapter of the Association for Computational Linguistics. 2010: 109-117.

[11] Trask A, Michalak P, Liu J. sense2vec-a fast and accurate method for word sense disambiguation in neural word embeddings [J]. arXiv preprint, arXiv: 1511. 06388.

[12] Sang E F T K, Meulder F D. Introduction to the CoNLL-2003 shared task: language independent named entity recongnition [C]// Proc of Conference on Natural Language Learning at Hlt-naacl. 2003: 142-147.

[13] Levy O, Goldberg Y, Dagan I. Improving distributional similarity with lessons learned from word embeddings [J]. Bulletin De La Société Botanique De France, 2015, 75 (3): 552-555.

[14] Levy O, Goldberg Y, Ramat-Gan I. Linguistic regularities in sparse and explicit word representations [C]// Proc of the 18th Conference on Computational Natural Language Learning. 2014: 171-180.

[15] Melamud O, Dagan I, Goldberger J. Modeling word meaning in context with substitute vectors [C]// Proc of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2015: 472-482.

[16] 唐明，朱磊，邹春显. 基于 Word2Vec 的一种文档向量表示 [J]. 计算机科学, 2016, 43 (6): 214-217.

[17] Zheng X, Chen H, Xu T. Deep learning for Chinese word segmentation and POS tagging [C]// Proc of EMNLP, 2013: 647-657.

[18] Diebold F X, Mariano R S. Comparing predictive accuracy [J]. Journal of Business&Economic Statistics, 2012, 13 (1): 134-144

[19] David M W. EvaluTion: form precision, recall and f-measure to ROC, informedness, markedness & CORRE-LATION [J]. Journal of Machine Learning Technologies, 2011 (1): 37-63.